Qwen3-1.7B 大模型端侧部署实战

课程名称：大模型系统与应用
实验环境：Linux Server (CPU Only) / Python 3.10

学号：522025330171

姓名：金东旭

1. 实验背景与目的

随着大语言模型（LLM）的发展，将模型部署在手机、PC 等端侧设备上（On-Device AI）成为热门趋势。本实验旨在让学生掌握 LLM 从 PyTorch 训练框架迁移到 ONNX 通用推理框架的完整流程。 **实验核心目标：**

1. **环境搭建：**掌握无 GPU 环境下的大模型开发环境配置。
2. **模型导出 (Export)：**理解 PyTorch 动态图与 ONNX 静态图的区别，掌握通过“算子补丁 (Monkey Patching)”和“模型包装 (Wrapper)”解决导出兼容性问题的方法。
3. **模型量化 (Quantization)：**实现 INT8 静态量化，理解量化对模型体积和推理速度的影响。
4. **推理实战：**编写 Python 脚本，加载量化后的模型实现一个简单的对话机器人。

2. 实验准备

2.1 硬件要求

- **CPU**: 建议 4 核以上 (Intel/AMD，指令集支持 AVX2/AVX512 体验更佳)
- **RAM**: 建议 16GB 以上 (FP32 模型导出过程瞬间内存占用较大)
- **Disk**: 至少 20GB 可用空间

2.2 软件依赖安装

请在终端中执行以下命令，创建 Conda 环境并安装 CPU 版本的 PyTorch 及相关工具。

```
# 1. 创建环境

conda create -n qwen_onnx python=3.10 -y

conda activate qwen_onnx
```

2. 安装 PyTorch (CPU版，减小下载体积)

pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

3. 安装 HuggingFace 生态与 ONNX 工具链

onnxscript 是 PyTorch 2.x 导出器的新依赖

pip install transformers accelerate onnx onnxruntime onnxscript

4. (可选) 设置 HuggingFace 国内镜像，加速模型下载

export HF_ENDPOINT=https://hf-mirror.com

pip install -U huggingface_hub

2.3 下载模型权重

使用命令行工具将 Qwen3-1.7B 模型下载到本地 Qwen3-1.7B 文件夹，避免实验中途网络中断。

huggingface-cli download --resume-download Qwen/Qwen3-1.7B --local-dir Qwen3-1.7B

3. 实验步骤

3.1 步骤一：编写导出脚本 (Export)注意：由于 Qwen3 使用了较新的注意力机制实现（如 vmap）,需要特殊处理机制

运行以下脚本会产生导出的模型文件

python export_stdu.py

3.2 INT8 静态量化 (Quantization)

python quant_stu.py

为了在 CPU 上流畅运行，我们需要将 FP32 模型量化为 INT8。这需要准备校准数据集，让模型“预演”推理过程以确定量化参数。

3.3：推理对话 Demo (Inference)

任务：创建文件 chat.py, 编写一个简单的对话脚本，验证量化后的模型是否能正常“说人话”。

python chatbot_stu.py

实验观察

文件体积对比： 记录导出模型和量化后的具体文件大小，并计算压缩比（例如：1750MB / 6800MB ≈ 25%)。 效果观察：

1. 量化后的模型在 CPU 上生成文字的速度如何？（主观感受即可，如：每秒大概几个字？）

FP32：

```
User: 你好
Qwen: <think>
好的，用户发来"你好"，我需要回应。首先，要保持友好和亲切的语气。可以简单问候，并表达愿意帮助的态度。比如："你好！有什么可以帮助你的吗？"或者"很高兴认识你，有什么我可以帮你的吗？"

3. **询问需求**：在初步交流后，询问对方的具体需求或问题，例如：
   - "您
[speed] 3.25 token/s, ~5.89 chars/s


User: 南京大学
Qwen: <think>
嗯，用户问的是"南京大学"，我需要先确定他们具体想知道什么。可能他们想了解学校的基本信息、历史、学科优势，或者校园生活、学术资源等。

接下来，我可以根据不同的情况来组织回答：

1. **基本情况介绍**：可以简要说明学校的地理位置、规模、主要院系等基本信息。
2. **课程设置与教学
[speed] 3.22 token/s, ~6.12 chars/s


User: 522025330171金东旭
Qwen: <think>
Okay, let's see. The user provided a number: 023301718945. They want me to convert this into a base-16 (hexadecimal) number. Let me think about how to approach this.

First, I need to recall that each digit in the hexadecimal system represents powers of 16. So, for example,
[speed] 3.00 token/s, ~10.58 chars/s
```

2. 量化后的模型是否依然能生成通顺的中文？请在报告中附上 2-3 轮对话的截图。

int8：

```
(qwen_onnx) PS E:\Desktop\study_projects\A1> python chatbot_stu.py int8
[mode=int8] model=qwen3_int8.onnx
ONNX input names: ['input_ids']
输入 exit / quit / q 退出。也可 Ctrl+C 退出。

User: jdx
Qwen: ACIÓNITOหมายITICALUGHT_DIRECTIONIOUSACTERIFIโฟIFFOTAIDADEILINGCENTITICALSTRACTỦSTIONشرイトTEDเตร之后 divingราวYZ
ŨIPHERfterOULDCTORLOOR党和国家IFT graduatingSTRACTSTRACT_ACCESSroneเหมUtó才发现沪指 bứcpoнجو_BOOLEANターUGHTCED_WEIGHT
UREOPSUGHTOUNDOUGH_SEGMENT_INSERT我才ITIES级IMATIONSTRACT_PACKAGESErrorSTRACTOUGHIMAL_VALIDATEギstanbulOOSE吒_RANKราวิ
ก tử
[speed] 7.76 token/s, ~32.21 chars/s

User: 南京大学
Qwen: 外地tóURELTEIFORMشرวลาราว调查显示IORLINGIDADEFFE graduatingターQUENCEOUGHYZMBER_GENERALSTRACTатopープ rpaфشرCED
IFTイトเดUDIOเตรราวagusATIVEỦSError二季度شرSTRACTึกราวNCLUSIONราว trậnSTRACTNGTH_RANKสมAMENTDVDIDADEشرOULDOUGHstanbul__
INCREMENTIPHERLECTIONUGHTITIONSSTRACTイルCEPTITICALIMATION tụTEDURTUREERGYADOW래STRACTราวSTRACTOTASTRACTughter
[speed] 7.40 token/s, ~29.87 chars/s
```

思考题：

在任务三中，目前的推理逻辑每生成一个字都要重新计算整个序列。如果想要加速，应该引入什么机制？（提示：KV Cache）

每生成一个 token，都把“整个序列 input_ids”喂进模型，模型会重新计算所有层对过去 token 的 attention（开销随序列长度增长）

KV Cache 机制的核心是：

第一次前向时，把每一层的 Key/Value（K/V） 保存下来（cache）

下一步只输入“新生成的那个 token”，并把过去的 K/V 作为 past_key_values 传回模型，模型只需要算新 token 的 K/V 并和历史缓存拼接，避免重复计算历史部分。在 ONNX/ORT 里要做 KV Cache，一般需要：模型图支持 输入 past_kv、输出 present_kv 自回归循环里维护并传递这些 kv 张量